PR9

Password Door Security

Version 1.2

Aug 2008

Cytron Technologies Sdn. Bhd.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Cytron Technologies Incorporated with respect to the accuracy or use of such information or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Cytron Technologies's products as critical components in life support systems is not authorized except with express written approval by Cytron Technologies. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

OVERVIEW

This document describes the development of Cytron Technologies DIY (Do It Yourself) Project No.9 (PR9). This project will use PIC16F877A, LCD screen and keypad to develop a password door security system. The system will activate the relay and buzzer if the password keyed in which is preset in the program is correct. Circuit schematic and PIC source code will be provided.

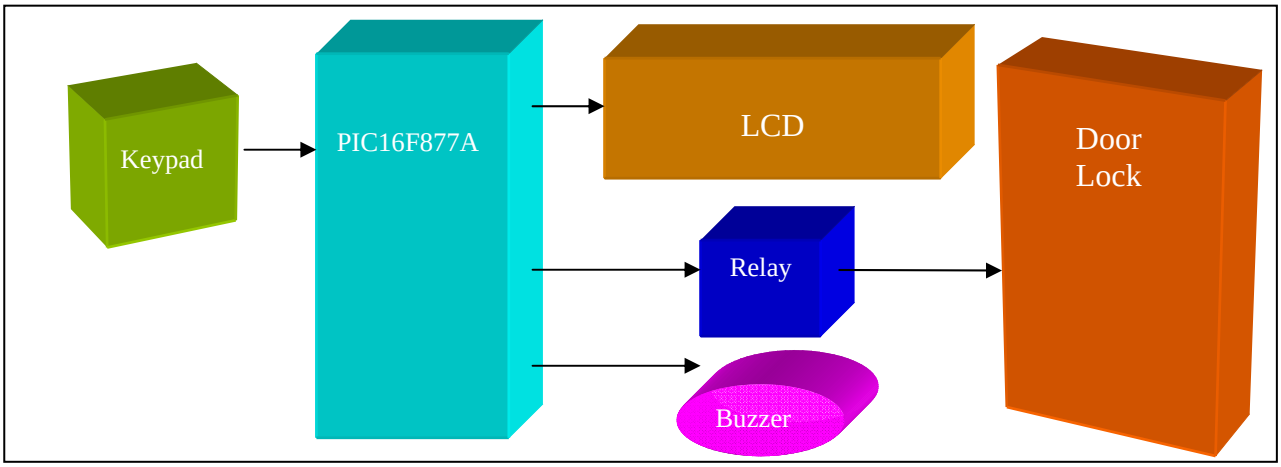
FEATURES

Basic circuit
- PIC16F877A as microcontroller
- Protection against wrong polarity on input power
- On board 5V voltage regulator (1A maximum)
- LED as power indicator
- 20MHz crystal
- Reset button
- ICSP provided for easy programming
- 4x3 keypad for key in password

LCD screen display
- Display words and password symbols
- Adjustable word contrast using potential meter

Relay and buzzer
- Output of the system
- Can be applied to any daily application such as door lock
- 2 pins port ready for application connection

SYSTEM OVERVIEW



GENERAL DESCRIPTION

This project shows the basic of developing a simple password door security system using microcontroller and can be further developed for more advance application.

PIC16F877A

This powerful (200 nanosecond instruction execution) yet easy-to-program (only 35 single word instructions) CMOS FLASH-based 8-bit microcontroller packs Microchip's powerful PIC® architecture into an 40- or 44-pin package and is upwards compatible with the PIC16C5X, PIC12CXXX and PIC16C7X devices. Features of the device:

- 256 bytes of EEPROM data memory
- Self programming
- ICD (In Circuit Debugging function)
- 2 Comparators
- 8 channels of 10-bit Analog-to-Digital (A/D) converter
- 2 capture/compare/PWM functions
- Synchronous serial port can be configured as either 3-wire Serial Peripheral Interface (SPI™) or the 2-wire Inter-Integrated Circuit (I²C™) bus
- Universal Asynchronous Receiver Transmitter (UART).

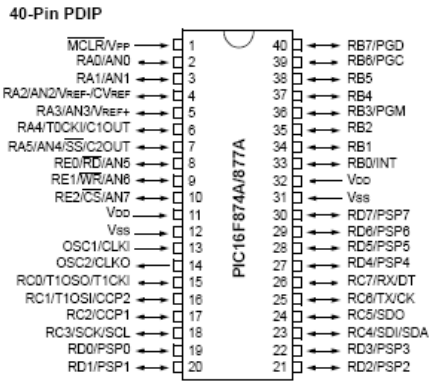


Figure 1

Figure 1 shows the pin diagram of the PIC16F877A. For more detail, please download the datasheet from microchip web site at: http://www.microchip.com

4X3 keypad

Figure 2

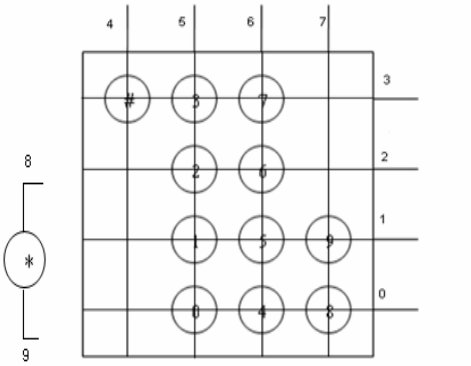


Figure 3

Figure 3 shows the internal structure of the 4x3 keypad used in this project. Different brand or manufacturer of keypad has different architecture and number of pin. For this kind of keypad, it consists of 10 pins and the internal connection is illustrated in Figure 3. When button 3 is pressed, the pin 3 and pin 5 will short while the others are open. Pin 8 and pin 9 are only connected to the '*' key.

Created by Cytron Technologies Sdn. Bhd. – All Rights Reserved

Relay

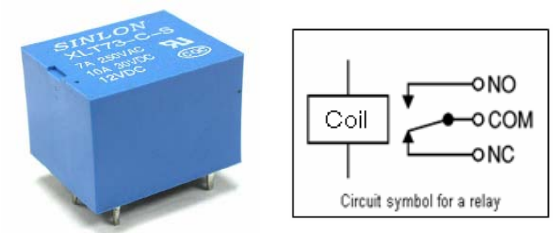


Figure 4

A relay (Figure 4) is a simple electromechanical switch made up of an electromagnet and a set of contacts. Current flow through the coil of the relay creates a magnetic field which attracts a lever and changes the switch contacts. The coil current can be ON or OFF so relay have two switch positions and they are double throw (changeover) switches. Relays allow one circuit to switch a second circuit which can be completely separate from the first. For example a low voltage battery circuit can use a relay to switch a 230V AC mains circuit. There is no electrical connection inside the relay between the two circuits; the link is magnetic and mechanical. The coil of a relay passes a relatively large current, typically 30mA for a 12V relay, but it can be as much as 100mA for relays designed to operate from lower voltages. Most ICs (chips) cannot provide this current and a *transistor* is usually used to amplify the small IC current to the larger value required for the relay coil. Relays are usually Single Pole Double Throw (SPDT) or Double Pole Double Throw (DPDT) but they can have many more sets of switch contacts, for example relays with 4 sets of changeover contacts are readily available.

Transistor

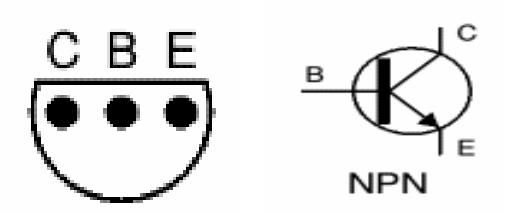


Figure 5

The transistor used to amplify the current for the relay is an NPN transistor, 2N2222A. The pin sequence is as shown in Figure 5. Beware that wrong sequence will burn the transistor.

Magnetic lock

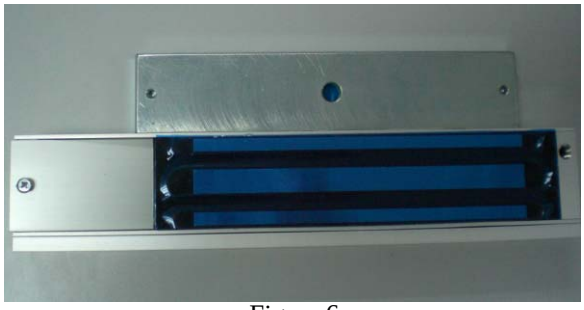


Figure 6

Figure 7

A **magnetic lock** is a simple locking device that consists of an electromagnet and armature plate. By attaching the electromagnet to the door frame and the armature plate to the door, a current passing through the electromagnet attracts the armature plate holding the door shut. Unlike an electric strike a magnetic lock has no interconnecting parts and is therefore not suitable for high security applications because it is possible to bypass the lock by disrupting the power supply. Nevertheless, the strength of today's magnetic locks compare well with conventional door locks and cost less than conventional light bulbs to operate.

HARDWARE

This project will require following hardware:

a. 1 x PIC16F877A
b. 1 x PR9 Printed Circuit Board (PCB)
c. 1 x 4x3 keypad
d. 1 x 2X16 LCD
e. 1 x relay
f. 1x 2N2222 transistor
g. 1x buzzer
h. Related electronic components

Please refer to the schematic diagram of PR9. The schematic is provided free and therefore Cytron Technologies will not be responsible for any further modification or improvement.

Created by Cytron Technologies Sdn. Bhd. – All Rights Reserved

Interface PIC with keypad

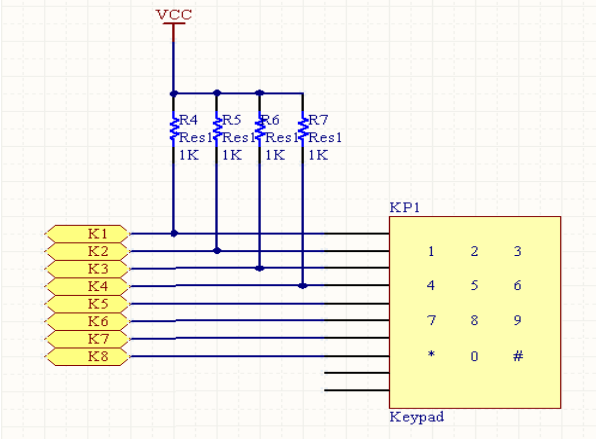


Figure 8

Keypad consists of 10 pins but in this project, the 9th and 10th pin are not connected because the key '*' are not in used (refer Figure 8). The 8 pins remaining are separated into 2 groups, 4 pins (K1-K4 in Figure 8) connect to the input of microcontroller and 4 pins (K5-K8 in Figure 8) connect to the output. User can decide any digital I/O pin for the input and output. Input must be pull high to 5V using a resistor and this configuration will result an active-low input.

Interface PIC16F877A with LCD (2X16 character)

To use the LCD, user has to solder 16 pin header pin to the LCD. LCD used in this project is JHD162A, for other type of LCD, please refer to its data sheet.

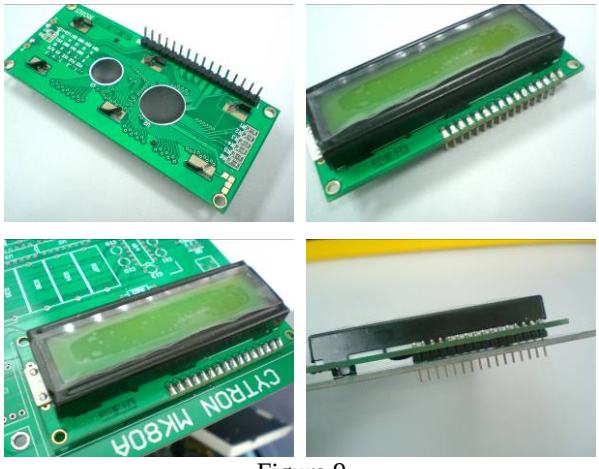


Figure 9

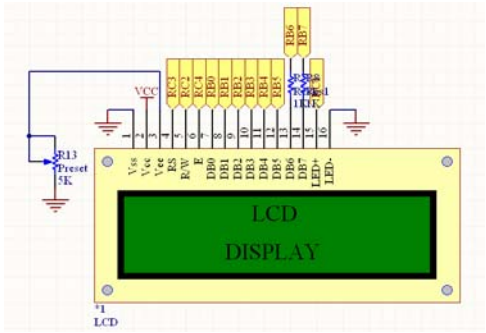


Figure 10

Figure 9 is a 2X16 character LCD. LCD connection pins and function of each pin are shown in table below:

Pin	Name	Pin function	Connection
1	VSS	Ground	GND
2	VCC	Positive supply for LCD	5V
3	VEE	Brightness adjust	Connected to a preset to adjust brightness
4	RS	Select register, select instruction or data register	RC0
5	R/W	Select read or write	GND
6	E	Start data read or write	RC1
7	DB0	Data bus pin	RD0
8	DB1	Data bus pin	RD1
9	DB2	Data bus pin	RD2
10	DB3	Data bus pin	RD3
11	DB4	Data bus pin	RD4
12	DB5	Data bus pin	RD5
13	DB6	Data bus pin	RD6
14	DB7	Data bus pin	RD7
15	LED+	Backlight positive input	VCC
16	LED-	Backlight negative input	GND

Power supply for the circuit

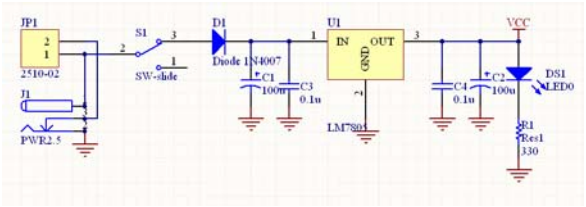


Figure 11

User can choose either use the AC to DC adaptor or 12V battery to power up the circuit. Higher input voltage will produce more heat at LM7805 voltage regulator. Typical voltage is 12V. Anyhow, LM7805 will still generate some heat at 12V. There are two type of power connector for the circuit, DC plug (J1) and 2510-02 (JP1). Normally AC to DC adaptor can be plugged to J1 type connector. Shown in Figure 11, the D1 is use to protect the circuit from wrong polarity supply. C1 and C3 is use to stabilize the voltage at the input side of the LM7805 voltage regulator, while the C2 and C4 is use to stabilize the voltage at the output side of the LM7805 voltage supply. DS1 is green LED to indicate the power status of the circuit. R1 is resistor to protect DS1 from over current that will burn the DS1.

Created by Cytron Technologies Sdn. Bhd. – All Rights Reserved

Relay as output of PIC microcontroller

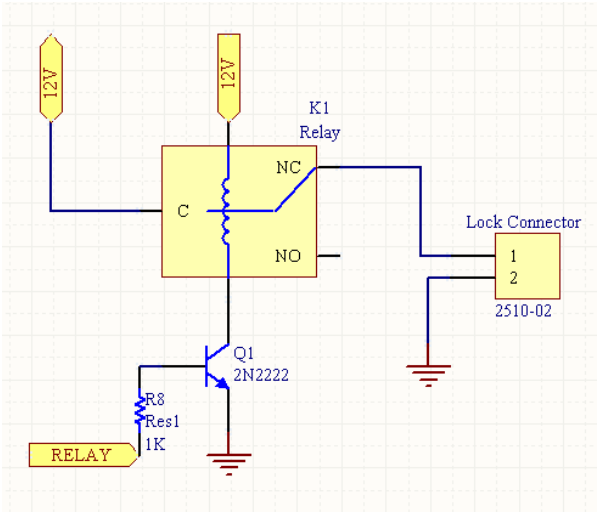


Figure 12

The relay used for the project consists of 5 pins, 2 pins is the 2 end of the coil, 1 is COM, 1 is NO and 1 is NC (refer Figure 12). One end of the coil is connected to 12V and another end is connected to an NPN transistor to amplify the small IC current to larger value required for the relay coil. COM pin is connected to 12V and NO is connected to a 2510 2-pin connector which is provided to locate the door magnetic lock.

Buzzer as output of PIC microcontroller

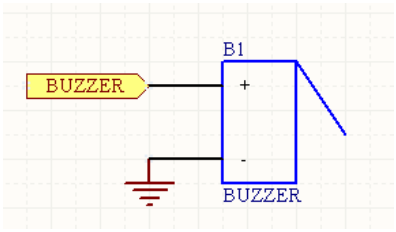


Figure 13

Buzzer can be connected to any I/O pin as output.

Push Button as input for PIC microcontroller

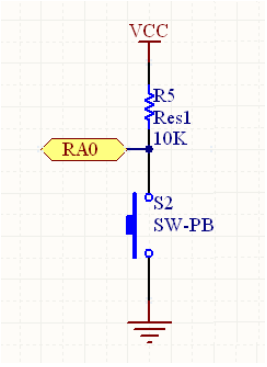


Figure 14

One I/O pin is needed for one push button as input of PIC microcontroller. The connection of the push button to the I/O pin is shown in Figure 14. The I/O pin should be pull up to 5V using a resistor (with value range 1K-10K) and this configuration will result an active-low input. When the button is being pressed, reading of I/O pin will be in logic 0, while when the button is not pressed, reading of that I/O pin will be logic 1.

LED as output for PIC microcontroller

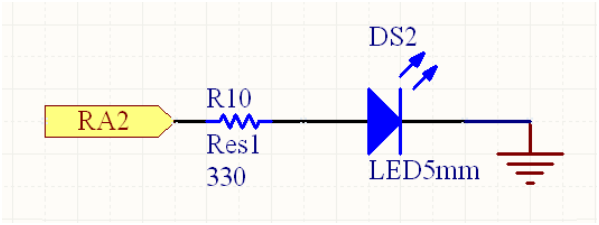


Figure 15

One I/O pin is needed for one LED as output of PIC microcontroller. The connection for a LED to I/O pin is shown in the schematic above. The function of R10 is to protect the LED from over current that will burn the LED. When the output is in logic 1, the LED will ON, while when the output is in logic 0, the LED will OFF.

ICSP for loading program

ICSP stands for In Circuit Serial Programming and describes the serial programming interface for PIC microcontroller. ICSP gives user a convenient way of programming PIC Microcontroller without removing the chip from the development or production board. User needs a programmer that provides the ICSP connector.

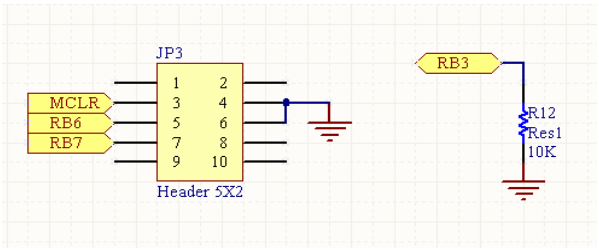


Figure 16

MCLR, RB6 and RB7 need to be connected to the ICSP box header to program the PIC microcontroller. At the same time, RB3 need to be pull down to 0V to disable low voltage programming, because the programmer is using high voltage programming.

Created by Cytron Technologies Sdn. Bhd. – All Rights Reserved

PCB circuit board

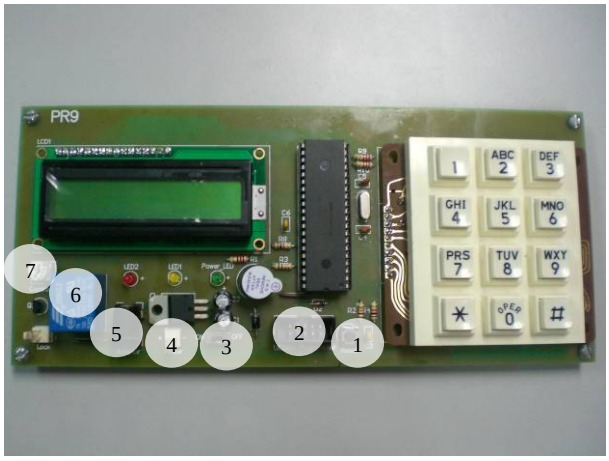


Figure 17

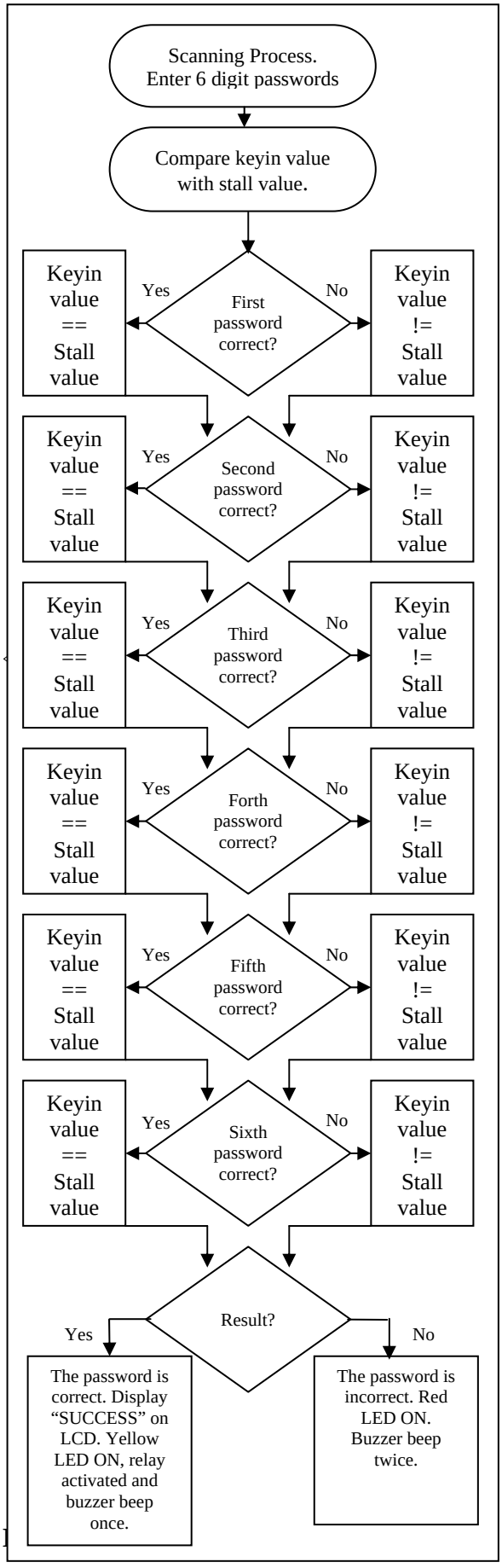
Component:

1. Reset button
2. Box header (To ICSP Programmer)
3. Slide switch (Power ON/OFF)
4. Power connector (12V)
5. DC plug socket (To 12V adaptor)
6. Application relay output (Motor)
7. Variable resistor (adjust LCD contrast)

Please refer to Appendix A for the PCB layout of PR9. The PCB layout is provided free therefore Cytron Technologies will not be responsible for any further modification or improvement.

SOFTWARE

Flowchart:


Scanning Process. Enter 6 digit passwords
Compare keyin value with stall value.
First password correct?
Second password correct?
Third password correct?
Forth password correct?
Fifth password correct?
Sixth password correct?
Yes
No
Keyin value == Stall value
Keyin value != Stall value
Result?
The password is correct. Display "SUCCESS" on LCD. Yellow LED ON, relay activated and buzzer beep once.
The password is incorrect. Red LED ON. Buzzer beep twice.

```
void main(void)
{
    ADCON1=0b00000110;  //set all portA pins as digital I/O
    TRISA=0b11001111;   //clear bit 4&5 portA as output and set the rest as input
    TRISB=0b00000000;   //set portB as output
    TRISD=0b00000000;   //set portD as output
    TRISC=0b11110000;   //set bit4-7 portC as input(connected to 4 row of keypad)
    TRISE=0b00000000;   //set portE as output

    PORTC=0;
    PORTD=0;
    relay=0;
    buzzer=0;
    led_yellow=0;
    led_red=0;
```

Listing 1

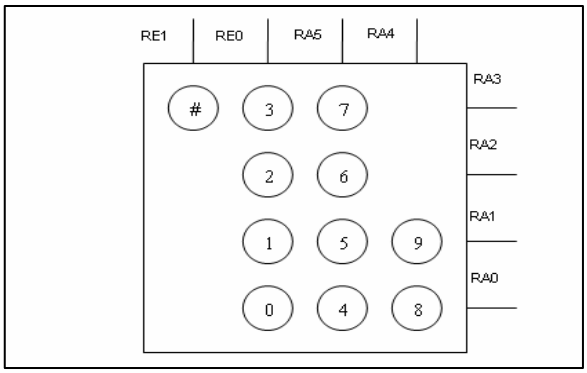
Listing 1 shows the configuration of the program and the initial value for certain variables and port pins. The ADCON1 register is configured with the binary value of '00000110' to change the entire portA pins into digital I/O instead of analog I/O. All the portA/B/C/D&E are bi-directional port meaning that they can be input or output. The corresponding data direction register are TRISA/B/C/D&E. Setting a TRIS bit (=1) will let the corresponding port bit an input. Clearing a TRIS bit (=0) will let the corresponding port bit an output. For an example, refer to the source code in Figure 1, TRISA is given the value '11001111' in binary. This means portA pin 4&5 are output while the rest are inputs. For further information, please refer to PIC16F877A data sheet.

```
send_config(0b00001001);    //clear display at lcd
send_config(0b00000010);    //Lcd Return to home
send_config(0b00000110);    //entry mode-cursor increase 1
send_config(0b00001100);    //diplay on, cursor off and cursor blink off
send_config(0b00111000);    //function

lcd_clr();
delay(1000);
lcd_goto(0);                        //initial display
send_string("PLEASE ENTER");
lcd_goto(20);
send_string("6-DIGIT PASSWORD");
```

Listing 2

The configuration of the LCD is shown in Listing 2. Any words can be sent and displayed by using the LCD function (refer to sample program). 'lcd_goto' function decides from which column the string start. For more information on the method to configure LCD, please refer to the data sheet.



Listing 3

```
while(1)
{                               //keypad scanning algorithm
        clearcolumn1();         //Clear 1st output pin and set the others
        scanrow1();             //scan row
        clearcolumn2();         //Clear 2nd output pin and set the others
        scanrow2();             //scan row
        clearcolumn3();         //Clear 3rd output pin and set the others
        scanrow3();             //scan row
        clearcolumn4();         //Clear 4th output pin and set the others
        scanrow4();             //scan row
```

Listing 4

```
void clearcolumn1(void)     //clear the 1st column and set the others
{
    RE1=0;                  //RE1,RE0, RA5 and RA4 are the
    RE0=1;                  //output pins from PIC which connect to 4 pins of keypad
    RA5=1;
    RA4=1;
}
void clearcolumn2(void)     //clear the 2nd column and set the others
{
    RE1=1;
    RE0=0;
    RA5=1;
    RA4=1;
}
void clearcolumn3(void)     //clear the 3rd column and set the others
{
    RE1=1;
    RE0=1;
    RA5=0;
    RA4=1;
}
void clearcolumn4(void)     //clear the 4th column and set the others
{
    RE1=1;
    RE0=1;
    RA5=1;
    RA4=0;
}
```

Listing 5

```
void scanrow1(void)
{
   if(RA3==0)                       //if key '#' is being pressed
   {
      while(RA3==0)continue;        //waiting the key to be released
      if(password_count==0)lcd_clr(); //Clear the LCD if the key is the 1st password
      lcd_goto(password_count);     //The cursor of LCD points to the column equivalent
      send_char('*');               //Display the symbol '*' at LCD
      keyin_char[password_count]='#'; //Stall the '#' value at the keyin_char array
      password_count+=1;            //increase the Password_count variable's value by 1
   }
}
```

Listing 6

```
void scanrow2(void)
{
    if(RA3==0)                          //if key '3' is being pressed
    {
        while(RA3==0)continue;          //waiting the key to be released
        if(password_count==0)lcd_clr(); //Clear the LCD if the key is the
        lcd_goto(password_count);       //The cursor of LCD points to the
        send_char('*');                 //Display the symbol '*' at LCD
        keyin_char[password_count]='3'; //Stall the '3' value at the keyir
        password_count+=1;              //increase the Password_count vari
    }
    else if(RA2==0)                     //if key '2' is being pressed
    {
        while(RA2==0)continue;          //waiting the key to be released
        if(password_count==0)lcd_clr(); //Clear the LCD if the key is the
        lcd_goto(password_count);       //The cursor of LCD points to the
        send_char('*');                 //Display the symbol '*' at LCD
        keyin_char[password_count]='2'; //Stall the '2' value at the keyir
        password_count+=1;              //increase the Password_count vari
    }
    else if(RA1==0)                     //if key '1' is being pressed
    {
        while(RA1==0)continue;          //waiting the key to be released
        if(password_count==0)lcd_clr(); //Clear the LCD if the key is the
        lcd_goto(password_count);       //The cursor of LCD points to the
        send_char('*');                 //Display the symbol '*' at LCD
        keyin_char[password_count]='1'; //Stall the '1' value at the keyir
        password_count+=1;              //increase the Password_count vari
    }
    else if(RA0==0)                      //if key '0' is being pressed
    {
        while(RA0==0)continue;          //waiting the key to be released
        if(password_count==0)lcd_clr(); //Clear the LCD if the key is th
        lcd_goto(password_count);       //The cursor of LCD points to th
        send_char('*');                 //Display the symbol '*' at LCD
        keyin_char[password_count]='0'; //Stall the '0' value at the key
        password_count+=1;              //increase the Password_count va
    }
```

Listing 7

Listing 3 shows the connection of the 4x3 keypad pins with PIC16F877A. When a key for example '7' is pressed, the 2 pin RA3 and RA5 will be shorted. Thus, to use a keypad without keypad decoder, the first eight pins of the keypad will have to be separated into 2 groups (4 pin to input and 4 pin to output of PIC). Refer to Listing 3, RA0-RA3 will set as input while RA4, RA5, RE0 and RE1 will set as output. Source code in Listing 4 shows a simple method to read the keypad. Program will scan row and column of keypad to read 6 digit password entered by user (Refer listing 5 and 6). First, clear the output pin RE1 and set the others. Go to a 'scanrow1' function shown in Listing 6. If the RA3 (input) detect a 0, it means the '#' key is pressed (please refer to the general description of keypad). Now, clear the second column which is pin RE0 and set the others. Go to 'scanrow2' function (refer listing 7) to scan whether the key '3', '2', '1' or '0' is being pressed. If RA1 (input pin) detects a 0, meaning that the key '1' is pressed. Clear RA5 and set the others and go for 'scanrow3' function to detect the key '7', '6', '5' or '4'. At last, clear RA4 and go for 'scanrow4' function. (Refer sample program)

Refer to the scanrow1 function in Listing 6, if RA3 pin equal to '0' ('#' key is pressed), the program under the "if" command will be activated. First, the "while" command will wait the '#' to be released to make sure the program under the "if" command will only run one time for a press. After that, the words on LCD screen will be cleared if the 'password_count' variable is equal to zero. The purpose of the 'password_count' variable is to let the LCD screen clear when the first digit is entered. The LCD will display the symbol '*' to tell user that the first digit is already been entered. Next, '#' value is stall at the keyin_char array. After that, password_count will increase 1 for 1 digit entered. Program will continue the process until password_count variable is equal to 6 or 6 digit password has been entered by user.

```
if(password_count==6)
{
    password_count=0;
    if((keyin_char[0]==stalled_char[0])&&(keyin_char[1]==stalled_char[1])&&
       (keyin_char[2]==stalled_char[2])&&(keyin_char[3]==stalled_char[3])&&
       (keyin_char[4]==stalled_char[4])&&(keyin_char[5]==stalled_char[5]))
    {
        lcd_clr();                      //clear lcd
        lcd_goto(0);
        send_string("SUCCESS!");        //display SUCCESS
        led_yellow=1;                   //green light on
        relay=1;                        //relay on
        beep_once();                    //beep one time
        while(1);                       //infinity loop
    }
    else
    {
        lcd_clr();                      //clear lcd
        lcd_goto(0);
        send_string("ERROR!");          //display ERROR!
        led_red=1;                      //red light on
        beep_twice();                   //beep two time
        while(1);                       //infinity loop
    }
}
```

Listing 8

After all 6 digits have been entered, program will compare value in keyin_char array with value in stalled_char array. Look at Listing 8, after the 'password_count' variable count to 6, the program will compare value in keyin_char array with value in stalled_char array. If value in keyin_char array is same with value in stalled_char array, LCD will display 'success', 'led_yellow' will ON, buzzer will beep once and relay will activated. If value in keyin_char array is different with value in stalled_char array, LCD will display 'error', 'led_red' will ON and buzzer will beep twice.

The source code is provided free and Cytron Technologies will not be responsible for any further modification or improvement.

GETTING START

User can obtain the hardware set for this project (PR9) either by online purchasing (www.cytron.com.my) or purchase it in Cytron Technologies Shop.

1. Once user has the hardware set, soldering process can be started. Please solder the electronic components one by one according the symbols or overlays on the Printed Circuit Board (PCB). Ensure the component value and polarity is correctly soldered. Please refer to PCB Layout in Appendix A.

Caution: Make sure all the connectors (2510) are soldered in proper side. Those electronic components have polarity such as capacitor, diode, PIC, LM7805 and LED should be soldered in right polarity or it may cause the circuit board fail to work.

Warning: Before the battery (Power) is plugged in, make sure the polarity is correct to prevent the explosion. Wrong polarity of capacitor also may cause explosion.

2. Please download the necessary files and document from Cytron Technologies website. These included documentation, sample source code, schematic, component list and software.

3. The next step is to install MPLAB IDE and HI-TECC C PRO into a computer. The MPLAB IDE and HI-TECH C PRO can be downloaded from www.cytron.com.my . Please refer MPLAB IDE installation step document to install the software. The documents can be used to any version of MPLAB IDE software.

4. After the installation complete, open the project file provided using MPLAB IDE. Please refer MPLAB Open Project document to open the sample program.

Created by Cytron Technologies Sdn. Bhd. – All Rights Reserved

5. Plug in power supply for the circuit. User can choose to use battery or AD to DC adaptor.

6. Build the project and load the hex file into the PIC microcontroller using the USB In Circuit Programmer (UIC00A). When user build the project, MPLAB IDE will generate hex file. The hex file generated from MPLAB IDE will be named according to project name, not C file name. Cytron Technologies also provide hex file for user. Do not forget to switch ON the power. The programmer is not included in the hardware set but it can be found at Cytron website. (User manual is provided at website).

7. This program can be modified. After modification, build the project and load once again the hex file into the PIC microcontroller using (UIC00A).

8. PIC is now completely programmed.

Remember! The default password is '123456'. If user wishes to change the password, some modification has to be done on the sample program. When the password key in is wrong, the red LED will light meanwhile if the password is correct, the green LED, buzzer and relay will ON until the reset button is pressed. User has to take note that this project is only display the concept of door lock system but for real life application, it need to be further modified.

AC to DC adaptor:

User can decide either uses a 12V battery or an AC to DC adaptor as the power source to the circuit. The picture and the way to use the adaptor are shown in Figure 18 and 19.

Figure 18 (not included in DIY project set)

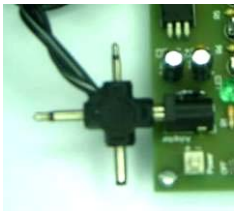


Figure 19

12V lead acid battery and connector:

Figure 20

Figure 21

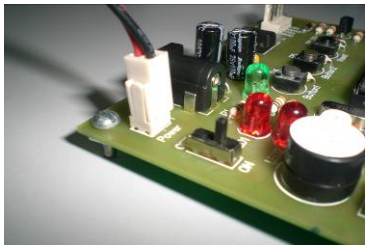


Figure 22

Figure 20 and 21 shows how to connect the cable leg to lead acid battery. If follow the standard, the red cable leg should be connected to terminal positive while the blue cable leg should be connected to negative terminal. Cable used to connect the cable leg and 2510 connector is provided in the project set. Red cable is connected to positive terminal and black cable is connected to negative terminal. Be careful on the polarity of the 2510 socket on the board (refer the PCB layout at Appendix A).

Created by Cytron Technologies Sdn. Bhd. – All Rights Reserved

How to connect the wire to 2510 connector:

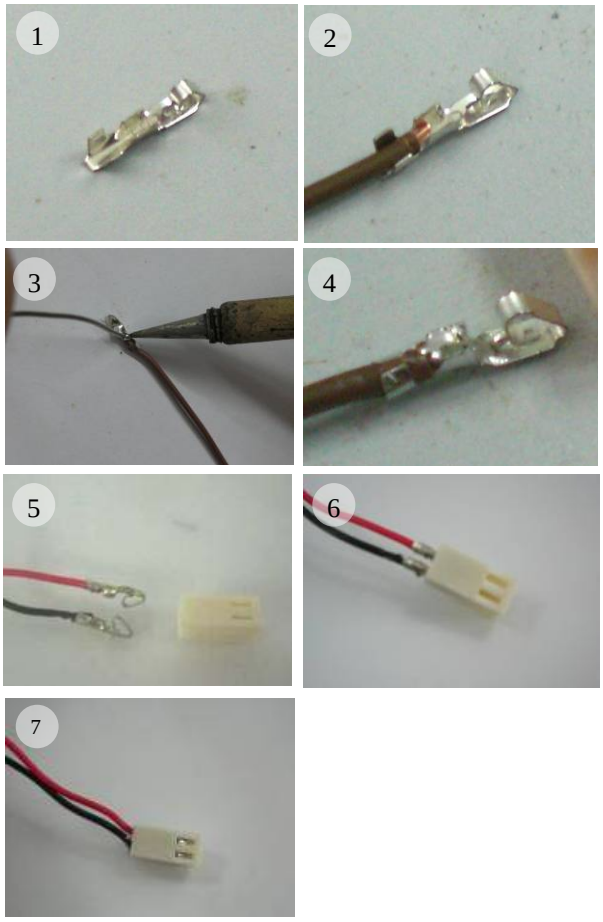


Figure 23

Figure 23 shows the method of connecting the cable to 2510 header.

How to wiring the magnetic lock:

1. Remove screw at magnetic lock like picture shown in Figure 24

Figure 24

2. Figure 25 shown magnetic lock after screw is remove.

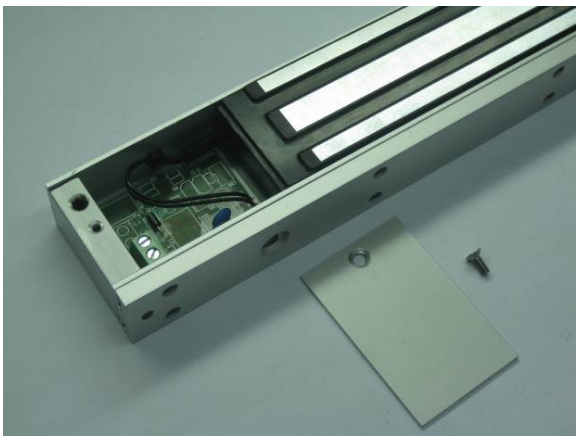


Figure 25

3. Make wire connection to 2510-02 Connector. Please refer “How to connect wire to 2510 Connector” in Figure 23.

4. Connect the other end of wire user build in step 3 to terminal block in magnetic lock. Magnetic lock doesn’t have polarity.

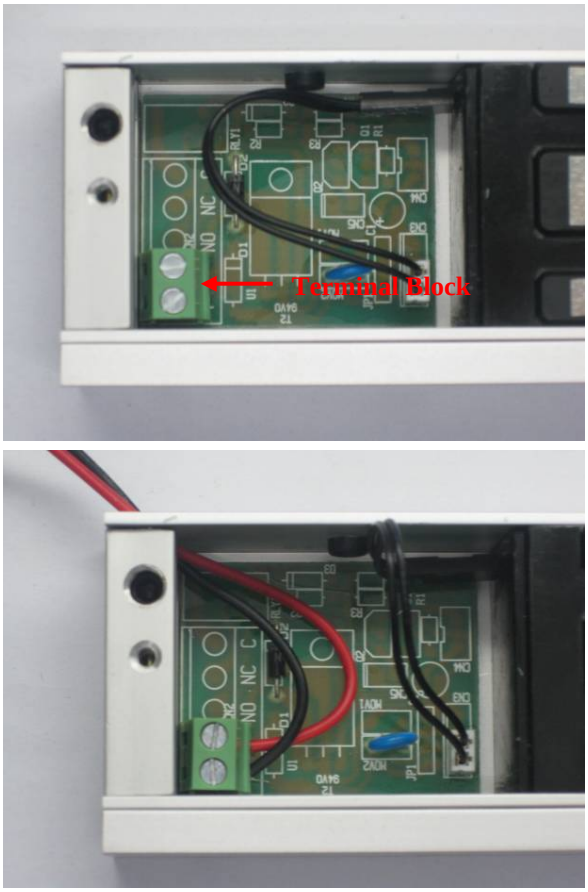


Figure 26

5. Screw back magnetic lock. Figure 27 shown magnetic lock with wire connector.

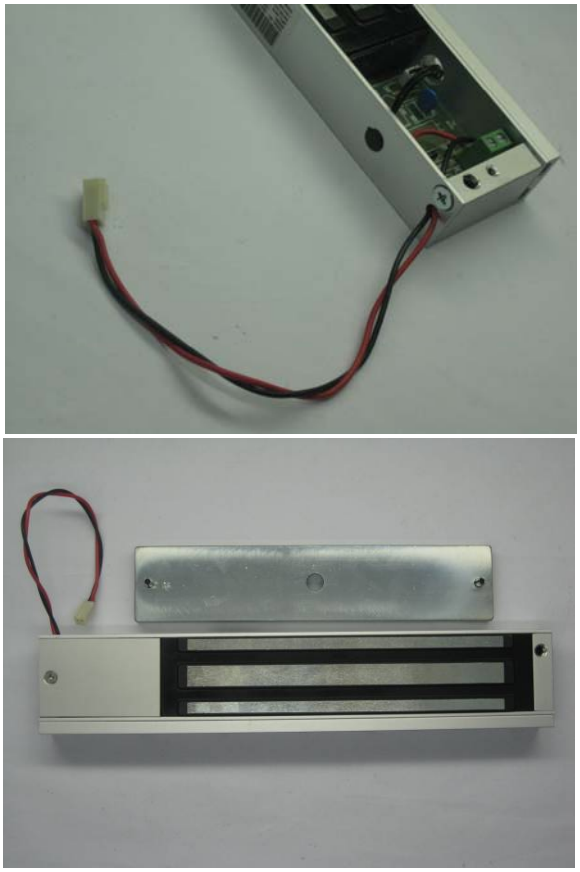


Figure 27

6. Connect magnetic lock to PR9 PCB board using 2510-02 connector.

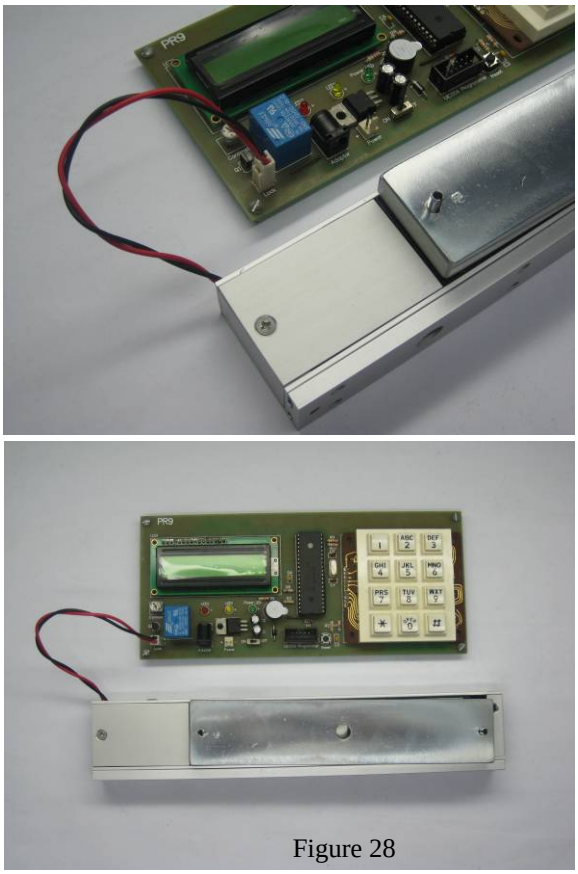


Figure 28

TEST METHOD

1. Switch ON the power
   - Power Led (Green) will turn ON.
   - LCD will display “PLEASE ENTER 6-DIGIT PASSWORD”
2. Press 123456 on keypad (default password)
   - LED1 (yellow) will turn ON.
   - LCD will display “SUCCESS!”
   - Door lock will release.
3. Press Reset button
   - LCD will display “PLEASE ENTER 6-DIGIT PASSWORD” again.
4. Press any 6 number on keypad (different with default password)
   - LED1 (red) will turn ON.
   - LCD will display “ERROR !”
   - Door lock will remain lock.
5. Press Reset button
   - LCD will display “PLEASE ENTER 6-DIGIT PASSWORD” again.
6. If all steps mention above can be executed, your project is done successfully. Congratulations!!

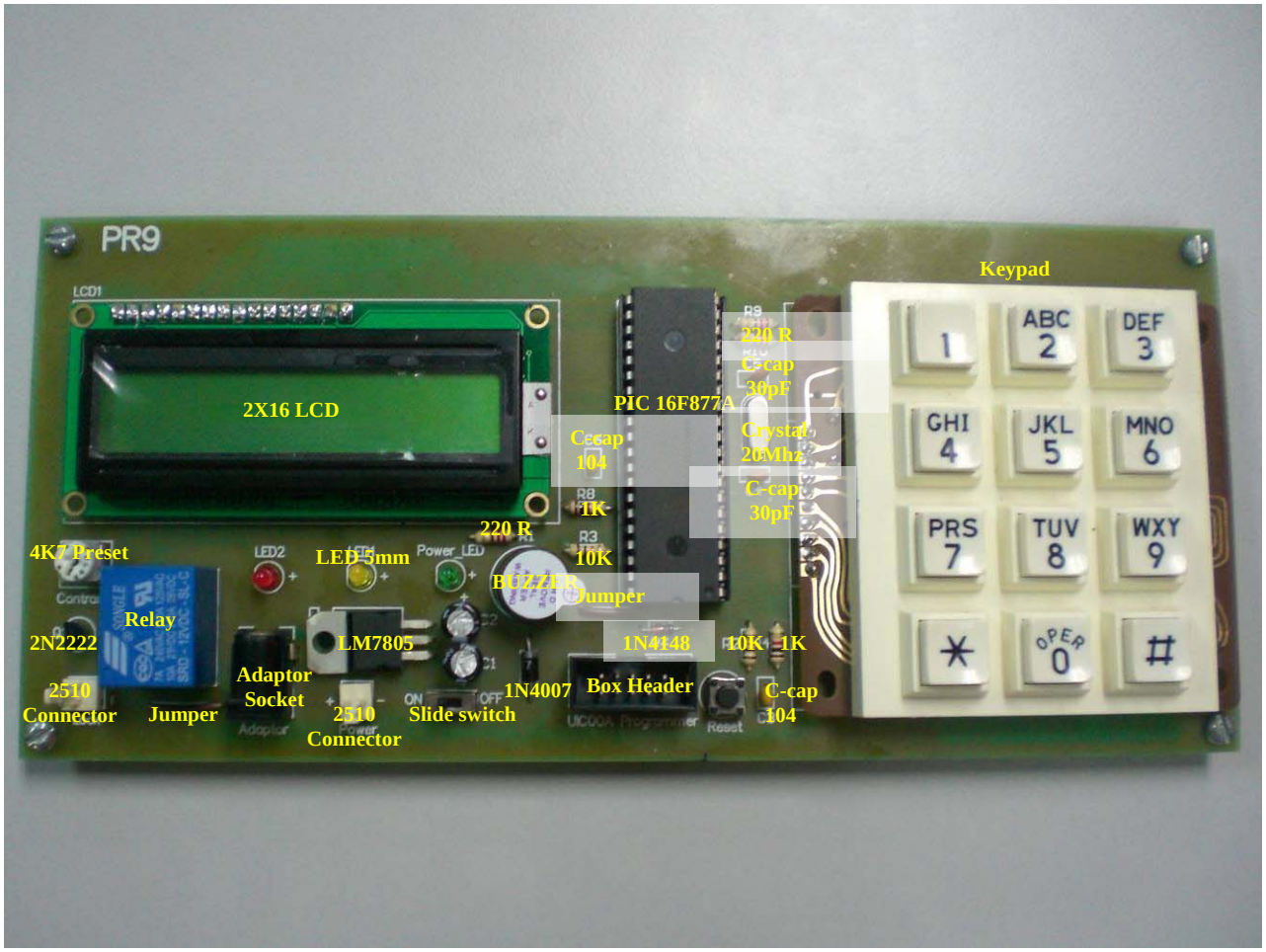
WARRANTY

No warranty will be provided as this is DIY project. Please check the polarity of each electronic component before soldering it to board.

Created by Cytron Technologies Sdn. Bhd. – All Rights Reserved

Appendix A

PCB Layout:



Prepared by
Cytron Technologies Sdn. Bhd.
19, Jalan Kebudayaan 1A,
Taman Universiti,
81300 Skudai,
Johor, Malaysia.

Tel: +607-521 3178
Fax: +607-521 1861

URL: www.cytron.com.my
Email: support@cytron.com.my
sales@cytron.com.my

Created by Cytron Technologies Sdn. Bhd. – All Rights Reserved